

Supersedes TM-2198/002/00

TM-2198/002/00

REPRINT

AD 712175

TECHNICAL MEMORANDUM

(TM Series)

This document was produced in connection with a research project sponsored by SDC's independent research program.

The MADAM System

by

William O. Crossley

2 December 1965

SYSTEM

DEVELOPMENT

CORPORATION

2500 COLORADO AVE.

SANTA MONICA

CALIFORNIA

Supersedes AD 456 940



for public release and distribution is authorized

Best Available Copy

2 December 1965

1
(Page 2 Blank)

TM-2198/002/00

ABSTRACT

MADAM (Moderately Advanced Data Management) is a program which builds and manipulates files, and generates reports. This document describes Version IV of the system--its rationale, utilization, grammar and vocabulary. Version I of the MADAM System, described in TM-2198/000/00, is superseded. All specifications prepared by reference to TM-2198/001/00 Version II manual, will operate properly in the Version IV system.

TABLE OF CONTENTS

	Page
INTRODUCTION.....	5
MADAM SYSTEM LANGUAGE CONVENTIONS.....	6
FILE DESIGNATIONS.....	8
SYSTEM OPERATIONS.....	9
DATA ENTITIES AFFECTED BY SYSTEM OPERATIONS.....	10
NON-PROCEDURAL SYSTEM OPERATIONS.....	11
LOAD.....	12
BLOCK.....	15
SORT.....	17
COPY.....	18
COMBINE.....	18
POSITION.....	19
OPERATE.....	20
MADAM LIBRARY OPERATIONS.....	21
PROCEDURAL SYSTEM OPERATIONS.....	26
PRINT.....	28
FORMAT.....	28
SUBSET.....	34
ABSTRACT.....	34
ACTION TERMS.....	35
LANGUAGE STRUCTURE TERMS.....	36
CONDITION TERMS.....	36
SPECIAL TERMS.....	38
DATA TERMS.....	40
DYNAMIC AND INDIRECT ADDRESSING.....	40
IMPLICIT TERMS.....	41
SPECIAL MADAM TERMS.....	42
USING THE MADAM SYSTEM.....	43
EXAMPLES.....	46
MADAM SYSTEM COMPUTER OPERATIONS.....	60
APPENDIX A.....	63

I. INTRODUCTION

MADAM (Moderately Advanced Data Management) is a file builder, a file manipulator, and report generator. It provides a means for taking initially unorganized lines of data (records) and organizing them in a describable and retrievable manner into data sets (files). It maintains these files, creates new ones, and prints the contents of these files in various formats. The system processes alphabetic and numeric data as well as English language text on the IBM 1401 computer. It was originally developed for use on metropolitan data problems; however, its data manipulation capability is quite general. First, the program translates the user's language to machine understandable language and communicates it to the machine. This process occurs once for each set of specification statements. Second, the machine interprets the translated path and performs the required operations once for each file record to be processed.

The MADAM System operates on a particular user's problem by means of a set of specification statements in a language very much like that of the user. MADAM itself is made up of computer programs which translate and interpret the specification statements and perform the required operations. The specifications are incorporated in a control card deck that is operated upon by the system in conjunction with data cards or magnetic tape files.

MADAM is implemented on an IBM 1401 computer with 8,000 characters core storage and four to five tape units. Its capabilities, representing only a state of moderate advancement in file processing and report generation, were selected because: they represent an area of high work volume and general need; they could be implemented within a reasonable time span; and they were possible within the space limitations of a small computer.

This guide to the use of the MADAM System describes in detail each MADAM operation in Chapters II and III. A brief summary of how the system may be used as well as examples are provided in Chapter IV.

II. MADAM SYSTEM LANGUAGE CONVENTIONS

The following paragraphs describe the operation and control of the MADAM System in terms of the English words that may be used to direct the system's operation, the data processing functions performed by the system, and the kinds of data entities or objects upon which the system operates. Some definitions are in order if ambiguity and confusion are to be avoided. A word, as defined in the MADAM language, is a string of alphanumeric characters separated from a preceding and following string by a blank or comma. A MADAM TERM is a word that has particular meaning for the MADAM System.

Any word composed of numeric characters only is a MADAM TERM. Any word having a left parenthesis as its leading character and a right parenthesis as its trailing character is a MADAM TERM. Any word in the following list is a MADAM TERM.

=	END	MORE
+	ENDA	NEXT
\$	ENDB	NQ
*	EQ	NUMBER
-	FIELD	NUMERIC
/	FORMAT	OPERATE
ABSTRACT	GQ	OR
ADJUST	GR	OUT
ALIGN	IDENT	PAGE
AND	IF	PAUSE
ANY	IN	POSITION
BLOCK	INA	PREFIX
BY	INB	PRINT
CALL	ITEM	READ
CARDS	KEY	READA
CATALOG	LIBRARY	READB
COLUMN	LINE	REMAINDER
COMBINE	LOAD	REPORT
COMPUTE	LQ	RESTORE
COPY	LS	RESULT
DO	MARK	SCAN

SET	SUBSET	WRITE
SIZE	SUFFIX	WRITE
SKIP	TEMP	**
SCRT	TERM	****
STOP	WORD	

Words occurring in MADAM System specifications, other than those enumerated above, are ignored by the system. The words above are also ignored except in the particular operation in which they exercise control.

MADAM specification cards are processed by the MADAM translator as a continuous stream of text starting with an operation declaration and ending with the MADAM term END. Between the END term of a previous specification and the operation declaration of the following specification, system control is returned from the operation level of translation to the system executive routine. To reduce control information transfer, the system executive expects to find the next operation declaration beginning on a card. For this reason, data following the terminal END on a MADAM specification card is not processed.

Within the continuous stream of text of a specification, words and parameters must be separated by at least one blank or comma. A period may not terminate a word unless that word is a label as defined in Chapter III Language Structure Terms. Any string of characters, including blanks, commas, periods, and left and right parenthesis may be declared to be a literal or constant term by preceding the literal string with the character pair--blank, left parenthesis--and by terminating the literal string by the character pair--right parenthesis, blank. A right parenthesis within a literal must not be directly followed by a blank or comma.

Data as understood by the MADAM System, include all keypunch characters legal for the IBM 1401 excepting the characters TAPE MARK (7-8), GROUP MARK (12-7-8), and RECORD MARK (0-2-8). These three characters, because of their special meaning in the IBM 1401 for magnetic tape handling, may not normally be included in data files but may be specified as constants or literals in the language. A TAPE MARK, however, may not appear among specifications catalogued on a MADAM library.

A. File Designations

FILES are designated by an ordered pair of numbers. The first number refers to the sequential position of the file on a reel of magnetic tape. The second designates the tape handler or unit on which a reel containing the file is mounted. One or more words may precede each number to assist readability by users. They are treated by MADAM as "noise" words and do not affect system operation. File designations may take the following forms in the case of file 2 on unit 3:

2,3

FILE 2 TAPE 3

FILE NUMBER 2 ON TAPE HANDLER 3

Two special file designations are also permitted:

- (1) FILE 0 TAPE 0, or the word CARDS, is interpreted to mean that the file is in IBM punch card form.
- (2) THE NEXT FILE ON UNIT 3, or NEXT 3, is interpreted to mean that no tape positioning is required to obtain the specified file.

Tape units which may be referred to in MADAM specifications are 0,2,3,4,5 and 6. Unit 1 is reserved for the MADAM System tape.

In MADAM System specifications output files are always declared prior to declaring input files. Thus the statement:

SUBSET 1,3 FROM 1,2

declares file 1 unit 3 as an output file and file 1 unit 2 as an input file.

When FORMAT, SUBSET, or ABSTRACT specifications call for two output files, the term AND is used to link the second output file declaration to the first and thus to differentiate it from input file designations that follow.

SUBSET 1,3 AND 1,4 FROM 1,2

declares file 1 unit 3 and file 1 unit 4 as output files.

III. SYSTEM OPERATIONS

The following MADAM TERMS initiate major operations in the system:

LOAD	Designates the operation of constructing a file on magnetic tape from punched card input. The input cards may be introduced via the card reader (direct) or via a previously prepared magnetic tape (prestored).
BLOCK	Designates an operation that permits records of a magnetic tape file to be grouped into collections of records. Each such collection becomes a single physical record in the output file.
SUBSET	Designates an operation in which records of a magnetic tape file may be selected for inclusion in one or two output files on the basis of complex selection criteria.
ABSTRACT	Designates an operation in which records from two magnetic tape files may be selected for combination into one or two output files based on complex selection criteria.
SORT	Designates an operation which orders the records of a magnetic tape file in alphabetic or numeric order by selected keys.
PRINT	Designates an operation in which the contents of a magnetic tape file may be displayed in the form of a printed report.
FORMAT	Designates an operation in which the contents of a magnetic tape file may be transformed and written as one or two new tape files and, simultaneously, included in a printed report.
COPY	Designates the operation of magnetic tape file reproduction. Binary tapes are not handled by the MADAM System.
COMBINE	Designates an operation in which files from an input magnetic tape may be specified for inclusion in a single output tape file.

OPERATE	Designates an operation in which the specified file is assumed to contain a 1401 program in AUTOCODER output unit record format: the program is loaded into core memory and control is transferred to the loaded program.
CATALOG	Designates an operation in which a named card-set (including but not limited to MADAM specifications) is loaded (catalogued) onto a library tape.
CALL	Designates an operation in which a named card-set is retrieved from a library tape. If the CALL operation occurs within a card-set which is being catalogued on the library, the named card-set replaces the card containing the CALL term. Otherwise the named card-set is assumed to contain MADAM specifications and is translated and executed.
POSITION	Designates an operation in which up to three magnetic tape units may be positioned to the starting record of specified files. This operation can be used for rewinding tape at the end of a complete job.
****	Introduces a message card. The contents of the card will be logged on the printer followed by a system pause.
**	Brackets a string of alphanumeric characters intended as a comment. The bracketed string is not translated

Data Entities Affected by System Operations

System operations may be performed upon logical entities and physical entities;

FIELD	A field is a succession or "string" of contiguous alphabetic or numeric (alphanumeric) characters. A field in the MADAM System may be up to 132 characters in length.
RECORD	A record is a collection of fields containing information relevant to an individual entity or thing. The standard maximum record size in MADAM is 1000 characters.

FILE	A file is a collection of records having a common logical structure and containing information relevant to a particular set of individual entities or things. In the MADAM System, a file may not exceed the storage capacity of one magnetic tape reel.
TAPE	A tape (reel of magnetic tape) may contain one or many files separated by end-of-file marks, each file composed of physical tape records separated by end-of-record gaps. Only BCD data is processed by the MADAM system.
CARD	A deck of IBM cards may contain a file. Each record in such a file may occupy one or many physical cards. Multicard records are identified by a card-set identity field in each such card. Card files are terminated by a card containing a GROUP MARK character (12-7-8 punch) in card column 1.
REPORT	A report represents the contents of a file displayed in printed form. Reports are made up of LINES and PAGES. A record may be represented in one or many lines or pages. Reports may be headed or formatted in any manner.

A. Non-Procedural System Operations

Operations in the MADAM System are specified with information files as their object.

The operations COPY, COMBINE and SORT require only little detail of the user as their processes are fairly standard to information handling.

The processes LOAD and BLOCK recognize fields within the unit records of the files they operate upon. Thus, data about where these fields are to be found in the input file and where they are to be positioned in the output file are required.

The processes PRINT, FORMAT, SUBSET, and ABSTRACT, in order to be useful across a broad spectrum of data handling problems, permit the user to express much greater detailed information about the files being handled or created. The majority of the power for logical manipulation of information is in these processes. Most of the MADAM language terms are needed here.

In the following paragraphs the use of each of the major MADAM processes will be described. The effect of MADAM TERMS applicable in each process will be shown in detail.

1. The LOAD Operation

The LOAD operation provides several standard methods of file building from 80-character unit records. The first method functions as a normal pre-store operation. Each incoming card or record occupies one record in the resulting file. A second method permits the user to specify card-set identification fields in the input unit records. In this option, all sequential cards having the same set identity will occupy the same record on tape. Card numbers are generated by the system according to their sequential occurrence. The third method is a refinement on the second. In addition to a card-set identification field for each card, a card number within a set may be specified. Thus cards need not be ordered sequentially within a set in this method. Each card will be uniquely identified by its identity field and number field. In any of the three methods, fields may be declared within any card and each declared field may be loaded at a specified position in the resulting record. A LOAD operation is initiated by a declaration in the form:

LOAD (output file designation) (input file designation)

The following examples represent forms of the LOAD declaration that are all acceptable to the system:

LOAD FILE 1 TAPE 2 FROM FILE 3 TAPE 3

LOAD FILE 1 TAPE 2 FROM CARDS

LOAD 1,2 FROM 3,3

LOAD 1,2,3,3

Only four English words, IDENT, FIELDS, NUMBER, and END, are understood by the LOAD operation. All others are considered to be "noise" words.

IDENT causes the program to expect two following numeric identifiers. The first such number is assumed to be the beginning card column of the field that contains a card-set identification. The field may be located anywhere in the card and may be of any size up to 80 characters. The second number is interpreted to represent the number of card columns in the card-set identity field. The use of the word IDENT is optional.

The card-set identity is used to group data from more than one card into the same record on the output tape. Cards having the same identity are considered to belong to the same record. For example, cards concerning an employee could be grouped according to his unique employee number.

NUMBER causes the program to expect two numbers specifying the beginning card column and number of columns containing a card identity number. The card identity number is used to associate the input card with the correct FIELD description as explained below. When card-set identities have been specified through use of the term IDENT, the use of the term NUMBER is optional. When IDENT is not used in the specification, the term NUMBER will be ignored.

FIELDS causes the program to expect sets of four numeric indicators to follow, describing fields to be loaded:

- a. The first number is the identity or sequence number of the card within a card-set. When no card identity number is supplied, cards within sets are automatically numbered sequentially from "1." Where no card-set identity is specified, card-sets are assumed to exist consisting of one card per set. The sequence number 1 must still be provided, even for a one-card set.
- b. The second number is the lowest numbered card column occupied by the field being described (1-80).
- c. The third is the number of card columns occupied by the field (1-80).
- d. The fourth and last number is the relative position in the tape record under construction that the left-most character of the field is to occupy. The first such position is location zero.

Each field (or string of contiguous characters) that is to be transferred from an output card to the tape record must be specified in the above manner.

END as a 3-character word standing alone terminates the LOAD specifications and initiates the LOAD operation.

A complete set of LOAD specification cards are presented below. Essential terms are underlined. All words, other than the three control words previously described, are noise words that have been included to provide more readable specifications.

LOAD FILE 1 TAPE 2 FROM CARDS

CARD SET IDENT IS IN COLUMN 77

FOR 4 COLUMNS

CARD NUMBER IS IN COLUMN 75 FOR 2 COLUMNS

FIELDS TO BE INCLUDED ARE AS FOLLOWS

CARD 1 START 1 SIZE 10 AT 0

CARD 1 START 21 SIZE 30 AT 10

CARD 2 START 1 SIZE 30 AT 40

CARD 3 START 65 SIZE 5 AT 70 END

In the above example, each record on tape would contain 40 characters from card number 1 at positions 0 thru 39 and, if cards 2 and 3 were present, 30 characters from card 2 at position 40 thru 69 and 5 characters from card 3 at position 70 thru 74.

If cards 1 or 2 were missing, positions 0 thru 49 or positions 40 thru 69 would contain blanks. If card 3 were missing, the record would be truncated at position 69. If cards 2 and 3 were missing, the record would be truncated at position 39.

If the optional terms NUMBER 75,2 in the example were not provided, and cards 1 and 2 were missing or out of order, the first card of the identified set that was encountered, in this case card 3, would be treated by the program as if it were card 1, the second as card 2 and so on. If the optional terms IDENT 77,4 in the example were not specified, each card encountered would be treated as card 1.

Input to the LOAD operation is limited to 80-column punch cards either via the card reader or from a previously stored magnetic tape.

Output record size is adjusted by the system to accommodate the fields that have been specified. The standard maximum record size accepted by all functions of the MADAM System is 1000 characters; however, records up to 4000 characters may be generated by MADAM to provide input to other systems or programs. When the NUMBER option is not used, data cards that are to be input to the LOAD operation must be ordered within any card set to assure correct loading. An end of file card (GROUP MARK character in column 1) must terminate the deck.

2. The BLOCK Operation

The BLOCK operation provides a means of reorganizing and restructuring a data file on magnetic tape. A structured file is first ordered with respect to desired key fields via the SUBSET and SORT operations. The key field is identified to the BLOCK process as the KEY data set identity field. Data fields from the old file that are to be included in the new file are identified to the BLOCK operation as data ITEMS. The BLOCK operation will group selected fields from records in the old file that have the same KEY into a single record in the new file. The data set identity field will be included as the first field in the new record. In order to provide user control of the maximum-size record that will be generated, a parameter SIZE is provided as described below.

The BLOCK operation may be used in a more straightforward way to group unrelated data elements into larger physical records by specifying only the maximum SIZE and ITEMS to be included, leaving the data set identity KEY unspecified.

The BLOCK operation is initiated by a declaration in the form:

BLOCK (output file designation) (input file designation)

Example:

BLOCK FILE 1 TAPE 4 FROM FILE 5 TAPE 2

BLOCK 1,4 FROM 5,2

Cards are not a permissible input form to the BLOCK operation. Four English words modify the operation of the BLOCK function:

KEY introduces two numeric indicators. The first number is interpreted as the left-most relative address (or lowest-valued address) occupied by a data set identity field. The second specifies the number of character positions occupied by the data set identity field. Records having the same data set identity are grouped together in a single blocked record up to the maximum specified block size. The data set identity KEY will be included in the output record occupying relative position zero through as many characters as are required. The use of the term KEY to effect data set identity blocking is optional. Data records will be blocked independently of content if no reference to KEY is made.

SIZE introduces a single numeric indicator that specifies the maximum number of character positions to be included in any blocked record. Specified items or fields from each input record will be included in the block being prepared until the addition of the data from a subsequent input record would cause the SIZE parameter to be exceeded.

ITEMS introduces recurring pairs of numbers that specify the relative starting address (left-most character position) and size (number of characters) for each item or field of the input record that is to be included in the output blocked record.

END as a 3-character word standing alone terminates the BLOCK specifications and initiates the BLOCK operation.

A sample set of BLOCK specifications are presented below. Significant terms and parameters are underlined. All other words are noise words whose only purpose is to enhance readability.

BLOCK FILE 2 TAPE 2 FROM FILE 3 TAPE 4

TO SIZE 2450

THE DATA SET KEY OCCUPIES POSITION

76 AND IS 4 CHARACTERS LONG

ITEMS TO BE INCLUDED ARE

START 0 SIZE 10

START 23 SIZE 24

START 50 SIZE 13 END

Each block resulting from the example specification will contain a data set KEY at position 0 thru 3, and as many following 47-character sets of input data as possess the same value of the data set KEY. The maximum block would include 52 such input sets.

Input to the BLOCK operation is assumed to be from magnetic tape. Input records can be as large as 1000 characters. Output records can be as large as 3500 characters. Records of 1000 characters are maximum for subsequent MADAM operations, however.

3. The SORT Operation

The SORT process permits records within a file to be ordered in alphabetic and/or numeric sequence with reference to key fields that are specified by the user. See Appendix A for 1401 collating sequence.

The SORT operation is initiated by a declaration in the form:

SORT (input file description) BY (major key field) AND (minor key fields)

Example:

SORT FILE 2 TAPE 2 BY 0,5 AND 5,1 AND 6,10 AND 79,1 END

Input to the SORT operation is assumed to be a file on magnetic tape.

The following terms control the operation of SORT:

END as a 3-character word terminates the SORT specification and initiates the SORT operation.

BY introduces the first of up to five sets of parameters, the sets being connected by the term AND. The first character of such a set specifies the relative starting (or left-most) character position, in the records to be sorted, occupied by a field to be used as a sort key. The second parameter in a set states the number of characters that make up the key. A third numeric parameter within any parameter set is considered to represent the maximum size (number of characters) of a tape record to be sorted. This third parameter may occur in any parameter set, where more than one set is stated and need occur only once in any SORT specification. It is used by the first phase of the SORT process to allocate core memory for internal sorting.

When a third parameter, as above, is not provided by a user, 1000 characters are assumed as the maximum tape record size. Sort keys are specified in descending hierarchical order of data organization. The major sort key follows the term BY and so on. Up to five hierarchical sort keys may be declared in one SORT operation.

A sample SORT specification is presented below with significant terms underlined.

SORT FILE 4 TAPE 2 BY 27,20,80 AND 48,2 AND
25,1 AND 19,5 AND 67,3 END

Input and output records may be of variable length but must not exceed 1000 characters. Sort keys must occupy identical relative positions in all records. Output will consist of a file on magnetic tape in 1401 collating sequence of sort keys. The sorted file will be the first file on that tape. The unit selected for output will be the unit upon which the input was provided. In the above example, output will be on FILE 1 TAPE 2.

During the SORT operation the term PAUSE occurring anywhere between the terms SORT and END will be interpreted as a request for a system halt at the completion of the first or distribution pass and a message will be printed. Operation is resumed when the operator presses START.

4. The COPY Operation

The COPY operation provides capability for duplicating data files between reels of magnetic tape. The COPY operation is initiated by a declaration in the form:

COPY (output file) (input file)

For example:

COPY FILE 1 TAPE 3 FROM FILE 2 TAPE 2

COPY NEXT FILE TAPE 3 FROM NEXT FILE TAPE 2

COPY 1,3 FROM 2,2

There are no English terms that modify the COPY operation.

Input is from magnetic tape and may consist of variable length records up to 4000 characters in length. Output is to magnetic tape with similar limitations.

5. The COMBINE Operation

The COMBINE operation permits any number of files from a reel of magnetic tape to be placed on another reel of magnetic tape as one file. The COMBINE operation is initiated by a declaration in the form:

COMBINE (output file) (initial input file)

Example:

COMBINE FILE 2 TAPE 3 FROM FILE 1 TAPE 2

COMBINE 2,3 FROM 1,2

Input and output are for magnetic tape.

Two English words modify the operation of the COMBINE function:

SKIP introduces one or many numeric parameters. These are interpreted as sequential file identity numbers for files from the input tape that are to be excluded from combination. The use of SKIP is optional.

TERMINATE introduces a single numeric indicator representing the sequential file identity number of the last file from the input tape to be included in the output file.

Input and output files may consist of variable length records up to 4000 characters long.

COMBINE FILE 1 TAPE 4 FROM FILE 1

TAPE 3 SKIP FILES 2,4,6, TERMINATE

AFTER FILE 7

6. The POSITION Operation

Each of the MADAM operations described above position a specified magnetic tape to a sequential file prior to the process. It is sometimes convenient to perform magnetic tape file positioning independently. The POSITION operation is provided for this purpose.

One to three magnetic tapes on separate tape handlers may be positioned with a single POSITION operation.

For example:

POSITION 1,2 END

POSITION FILE 1 UNIT 2, FILE 3 UNIT 3 END

POSITION 1,2,1,3,1,4 END

The third and last example above can be employed to rewind three reels at the end of a MADAM job. The POSITION operation may be used to assign input-output units to an operation. In this case the POSITION specification is not terminated by an END term. For example:

POSITION 1,2 FROM CARDS

FORMAT

READ WRITE IN 0,80 END

will be interpreted same as:

FORMAT 1,2 FROM CARDS

READ WRITE IN 0, 80 END

The first form above can be used with the CATALOG and CALL operation described elsewhere in this manual to effect input-output unit assignment and re-assignment for the same specification.

There are no modifying words affecting the POSITION function.

7. The OPERATE Function

The OPERATE function may be used to load and execute any compatible M401 program.

Program features that determine compatibility with the MADAM system are:

- a. The program must be in AUTOCODER condensed card form, either as a deck or prestored as a file on tape.
- b. The program must be compiled to occupy core memory locations 354 and higher.
- c. The program must return control to the MADAM system through the following AUTOCODER instruction set or its equivalent:

RWD	1	rewind unit 1
RTW	1,1	read the first record from unit 1 into location 1
B	31	Branch to location 31

- d. The program must obtain object data from a file and place its output into a file. No provision is made for core memory storage of data to be communicated between programs.

When the program to be operated contains AUTOCODER clear core cards 1 and 2, the OPERATE function performs the clearing operation in a way that preserves MADAM operation parameters.

OPERATE specifications take the following form:

OPERATE (file declaration)

OPERATE CARDS causes a following AUTOCODER condensed program deck to be loaded and executed. OPERATE 2,4 causes the second file from tape unit four to be treated as a prestored AUTOCODER condensed program deck for loading and execution.

OPERATE NEXT, 6 may be used in conjunction with the CATALOG and CALL operations described elsewhere in the manual to place a non-system program into the system library and call it into execution by name.

B. MADAM LIBRARY Operations

The LIBRARY operations provide a capability for placing card sets into a library file. Each such card set is given a distinguishing name at the time of insertion. Named card sets may be retrieved for subsequent processing.

Card sets in the MADAM library may contain:

- (1) Complete MADAM specifications including compatible 1401 programs.
- (2) Comments.
- (3) Combinations of 1 and 2 above.
- (4) Partial MADAM specifications or comments (legal only in the CATALOG mode, see below).

LIBRARY operations occur in two modes:

- (1) The CATALOG mode in which card sets are retrieved for loading into a library file in association with an identifying card set name.
- (2) The EXECUTE mode in which named card sets are retrieved for translation and execution by the MADAM processes. In the EXECUTE mode, card sets are assumed to represent executable complete MADAM specifications, compatible 1401 programs, or groups of same.

The EXECUTE mode is the normal operating mode of the MADAM system. The CATALOG mode is entered when a CATALOG specification occurs.

The MADAM library consists of a single file containing sequential 80 character card records. The file is terminated by a single 1401 end-of-file marker.

The following operations may be performed on the library file:

- (1) The operation LIBRARY is used to initiate or open a library file. The LIBRARY specification takes the form:

LIBRARY (file declaration)

LIBRARY 1,6 opens a library file as file 1, unit 6. Tape unit 6 is rewound to load point and a single end-of-file mark written as the first record.

LIBRARY CARDS opens a library file in punched card form. No initializing actions are taken other than to identify the output form as punched cards. The LIBRARY operation is used only when a new library file is to be built. File 1 unit 6 is implicitly defined in the MADAM system as the operating library.

No LIBRARY specification is needed to add new card sets into the operating library on file 1, unit 6.

- (2) The CATALOG operation is used to insert a named card set into an existing library. The CATALOG specification takes the form:

CATALOG (card set name) where card columns 1-7 of the specification card contain the term CATALOG, card columns 9-18 contain a user specified unique card set name, and card columns 19-80 may contain any comment or descriptive text.

The CATALOG specification card is followed immediately by the card set to be catalogued. The object card set is terminated by a card containing MARK in card columns 1-4.

At the completion of a CATALOG operation the designated LIBRARY file contains:

- a. the CATALOG card record
- b. the card set as sequential card records
- c. the MARK card record

The records above will be added to the library file beginning at the current record position occupied by the end-of-file record. An end-of-file record will be written following the MARK card record.

3. The operation CALL is used to retrieve named card sets from a library file on tape unit 6. The library file must be the first file on unit 6.

The CALL specification take the form:

CALL (card set name)

Only one CALL specification per card is allowed. There is no requirement as to card columns excepting the normal requirement for a blank or comma between successive terms. The first ten characters only of a card set name are used for retrieval matching.

In the EXECUTE mode, the CALLED card set is submitted directly to the MADAM translator as executable MADAM specifications.

In the CATALOG mode, the CALLED card set is included in sequence within the total card set being catalogued. The CATALOG and MARK cards of the CALLED card set are not retrieved from the source library. It is possible, in the CATALOG mode, to thus combine selected partial MADAM specifications into a complete specification for later execution from tape 1 unit 6.

For example:

```
LIBRARY 1,6
CATALOG PRESTORE
FORMAT 1,2 FROM CARDS
READ WRITE IN 0, 80 END
MARK

CATALOG PROG-1
OPERATE NEXT, 6
(1401 compatible program deck)
MARK

CATALOG LIST
PRINT 1,4
READ SET IN 0, 80 AT OUT 0 PRINT END
MARK
```

In sequence above:

- a. A library file is opened on tape unit 6
- b. The job PRESTORE is entered
- c. The AUTOCODER program PROG-1 is entered (note it is the OPERATE next, 6 specification that actually effects program execution)
- d. The job LIST is entered

The user may then say:

CATALOG JOB-1.

CALL PRESTORE

CALL PROG-1

MARK

CATALOG JOB-1.-A

CALL JOB-1.

CALL LIST

MARK

Now, when the user requests:

CALL JOB-1.-A

All three of the original specifications above are retrieved for execution. Note that a CALL within the brackets CATALOG and MARK is not retrieved for execution but for inclusion into the library. Note also that LIBRARY 1,6 is only specified where no operating library is currently on-line to the MADAM System.

To copy the above library onto another file:

LIBRARY 2,4

CATALOG PRESTORE

CALL PRESTORE

2 December 1965

25

TM-2198/002/00

MARK

CATALOG PROG-1

CALL PROG-1

MARK

CATALOG LIST

CALL LIST

MARK

CATALOG JOB-1.

CALL JOB-1.

MARK

CATALOG JOB-1.-A

CALL JOB-1.-A

MARK

Any or all of the catalogued card sets could have been re-named
in the above example.

C. Procedural System Operations

The non-procedural processes described previously operate iteratively upon each record of the input or output file. There is a means of selecting object fields or records before the fact, but no method is provided in those processes for dynamic re-specification of object fields during operation as a result of changes in the object data. This is perhaps not too restrictive for the above operations and has the advantage that, because each element of data need not be examined for evaluation purposes, operation can proceed more rapidly.

There exists a class of operations upon data, however, that must take into account the dynamic nature of the data being processed. These conditional operations must be made to perform a designated task only when certain conditions are manifested as a result of data manipulation or examination.

In the processes to be explained below, MADAM provides a useful range of capability that can be made conditional upon both the logical and physical nature of the data as it is dynamically encountered.

Conditional operation in MADAM is effected through the implementation of two kinds of functions: tests and conditional actions. Neither function need be particularly extensive in its range of operation. The following tests can be made in MADAM.

- (1) Tests for end of input files.
- (2) Tests for end of variable length fields such as textual fields and textual elements.
- (3) Tests to determine the nature of fields:
 - a. Tests for a numeric field;
 - b. Tests that determine the relative collating sequence of field configuration for two fields;
 - c. Tests that determine the relative collating sequence of the values of two fields.
- (4) Tests that permit format control of printed output.

The following kinds of conditional actions are provided:

Reading and writing of data files from various sources;

Transfer of data fields within internal working areas;

Editing actions that permit fields to be formatted in particular ways;

Arithmetic actions upon fields;

Actions to control the direct printer carriage;

Actions to control the sequence of execution of tests and other actions;

Actions that permit dynamic definition of object fields.

The four MADAM operations in which tests and conditional actions can be taken are: PRINT, FORMAT, SUBSET, and ABSTRACT. In SUBSET, ABSTRACT, and FORMAT, two output files may optionally be produced.

1. The PRINT operation is used when the contents of a data file are to be displayed in printed form as in a report. The data file to be printed may be sorted on magnetic tape or may be in card form. The format of a report prepared by the PRINT operation is flexible due to the nature of the conditional operations employed.

The PRINT operation is initiated by a declaration in the form:

PRINT (input file designation)

Examples:

PRINT FILE 2 TAPE 3

PRINT CARDS

The data to be printed is read into an internal input storage region referred to by the user as region IN. Each line that is to be printed is prepared in another storage region labelled OUT. Region OUT is large enough to contain a full 132-character line. The operations performed during a PRINT are described in detail by the user in a set of operation specifications directing that conditional actions be performed.

2. The FORMAT operation is used when the contents of a data file are to be written onto a magnetic tape or punched into cards. Like PRINT, the data file to be formatted may be stored on magnetic tape or may be in card form. During the FORMAT operation the full range of capabilities of the PRINT operation are available to the user. This means that combination operations may be done at the same time such as printing and punching or printing while writing records onto tape. While printing can be done only from region OUT (132 characters), tape records may be written from regions IN (1000 characters) or region OUT.

The FORMAT operation is initiated by a declaration in the form:

FORMAT (output file designations) (input file designation)

Examples:

FORMAT FILE 2 TAPE 3 FROM FILE 3 TAPE 4

FORMAT 2,3 FROM CARDS

FORMAT CARDS FROM CARDS

FORMAT 1,5 AND 1,3 FROM 1,2 (Where two output files are desired)

Like PRINT, the data from the input file is read into an internal storage region referred to as region IN. The data to be printed is placed into region OUT by conditional transfer action. Data to be written onto tape or punched into cards may remain in region IN or be transferred to region OUT optionally. The WRITE action is performed from a specified region. The full range of conditional operations explained in the following paragraphs may be employed in FORMAT or a PRINT operation.

PRINT and FORMAT Actions

Actions that may be taken during a PRINT or FORMAT operation are listed below with some explanation as to their effect on the object data.

ACTION

EFFECT

READ

The next record from the designated input file (tape or cards) is read into region IN beginning at relative position IN 0. A GROUP MARK (12-7-8 punch) is placed in region IN following the last data character read in. The COLUMN counter is incremented.

<u>ACTION</u>	<u>EFFECT</u>
READ 100	<p>The next record from the designated input file (tape or cards) is read into region IN at location IN 0. A check is then made to see if at least 100 characters are present in the region. If not, the next record is read in following the first and so on until 100 characters are present.</p> <p>On the next READ 100 action, any characters in excess of 100 that were previously read are saved at position IN 0 and the next record is read in following them and so on as above. The COLUMN counter is incremented.</p>
WRITE IN 0, 80	<p>The 80 characters beginning with the character at position IN 0 are transferred to the designated output file as the next record. Characters in region IN are not changed. GROUP MARK characters in region IN will terminate the WRITE operation.</p>
WRITB IN 0,80	<p>Similar to WRITE IN 0,80 except the second designated output file receives the transferred data. (The second output file designation is linked to the first by the term AND. Not legal during PRINT operation.)</p>
PRINT	<p>Data in region OUT 0 thru OUT 131 are printed as the next line on the direct printer page. Region OUT 0 thru 131 is cleared to blanks. The LINE counter is incremented.</p>
SKIP 1	<p>The next line on the direct printer page is bypassed and the LINE counter incremented.</p>
RESTORE	<p>This action starts a new page. The direct printer carriage is positioned to conform to a punch in channel 1 of the carriage control tape. The standard carriage control tape is so prepared as to position the paper to the top of the next page. The PAGE counter is incremented and the LINE counter set to 1.</p>

ACTIONEFFECT

REPORT 55

This is an action that permits the user to specify other than 60 lines per page. In the example, 55 lines are desired. An internal register is set to 55. If the LINE counter attains a value of 55 or more and the PRINT or FORMAT specifications permit, an automatic RESTORE action is taken.

NOTE: The automatic RESTORE will not function unless the PRINT or FORMAT specifications are written such that the END (end of specifications) term is executed. Should the user restrict the execution of the END term by interposed loops via the DO action, he must provide page control actions in his specifications.

SET (01) AT TEMP 1

The two characters enclosed in parentheses, "01," are transferred to region TEMP with the digit "0" occupying relative position TEMP 1 and the digit "1" relative position TEMP 2.

ADJUST (012) AT TEMP 1

The three characters enclosed in parentheses, "012," are examined and leading zeros and/or all blanks are removed from the field. The remaining characters, in this case the digits "12," are transferred to region TEMP such that the right-most character "2" occupies relative position TEMP 1 and the preceding character "1" relative position TEMP 0.

ALIGN (012) AT TEMP 1

Leading zeros and/or all blanks are removed as above. The remaining characters, "12," are positioned to region TEMP with the digit "1" at TEMP 1 and the digit "2" at TEMP 2.

STOP

An end-of-file is written on any output tape and return is made to MADAM System control where the card reader is monitored for a new specification.

ACTIONEFFECT

COMPUTE IN 0,5

The 10-digit arithmetic accumulator (RESULT) is cleared to zero and the 5 characters at IN 0 thru IN 4 are added as integers. COMPUTE with an object data field greater than 10 characters in length will cause standard 1401 overflow bits to be generated at position RESULT 0. COMPUTE is valid with a literal object such as COMPUTE (0). This would have the effect of clearing and adding (0) to the accumulator.

+ IN 0,3

The 3 characters at IN 0 thru IN 2 are added to the current contents of the 10 digit arithmetic accumulator (RESULT). Overflow may occur as in COMPUTE.

- (5)

The 1 character literal value "5" is subtracted from the current contents of the 10 digit accumulator (RESULT). If the result is negative the units position, RESULT 9, will have a minus (or 11 zone). A positive result will have no zone bits at RESULT 9.

* OUT 5,3

The current contents of the 10 digit accumulator (RESULT) are multiplied by the 3 character value at OUT 5 thru OUT 7. Overflow may occur.

/ IN 3,2

The current contents of the 10 digit accumulator (RESULT) are divided by the 2 character value at IN 3 thru IN 4. The integral portion of the quotient is left in the accumulator. Any remainder may be found in a 10 digit field (REMAINDER) as an integer.

\$ IN 3,2

The current contents of the 10 digit accumulator (RESULT) are divided by the 2 character value at IN 3 thru IN 4. The remainder is left in the accumulator as an integer. The quotient is destroyed.

= OUT 0,5

The 5 least significant characters of the current contents of the 10 digit accumulator (RESULT) are transferred to OUT thru OUT 4. The value in the accumulator is unchanged.

ACTIONEFFECT

= OUT 0,20

The 10 digits of current accumulator contents are transferred to OUT 10 thru OUT 19. The value in the accumulator is unchanged. The data characters at OUT 0 thru OUT 9 are not affected.

SCAN IN 0,80

The 80 characters at IN 0 thru IN 79 are identified as a textual field. Only one textual field may be scanned at a time; however, textual field identification may be dynamically changed during any process. On the first occurrence of the terms "NEXT WORD," a word will be sought beginning at IN 0 and proceeding towards IN 79 until a word separator character, or position IN 80, is reached. At this point, the string of characters isolated has its starting position and size identified. Subsequent reference to "WORD" will cause the identified data string to be operated upon.

The following regions may be referred to in the PRINT and FORMAT operations.

<u>REGION</u>	<u>SIZE</u>	<u>CONTENT</u>
IN	1000 char.	Contains an input record as a result of a READ action. May be used for other storage.
OUT	132 char.	May be loaded with data to be printed, punched, or written onto magnetic tape.
TEMP	200 char.	May be used for temporary storage. Contains index values for the prime operation.*
REMAINDER	10 char.	Contains remainder after a divide action.
TERM	6 char.	Positions 0 thru 4 contain up to five legal word separator characters as specified for the text scanning action. Position 5 contains the last word separator actually encountered during scanning.
RESULT	10 char.	Accumulates arithmetic results.

* The prime operation is explained on page 41.

<u>REGION</u>	<u>SIZE</u>	<u>CONTENT</u>
COLUMN	3 char.	Input record counter.
PAGE	3 char.	Page counter.
LINE	3 char.	Line counter.

3. The SUBSET operation is used when the contents of a data file are to be selectively transferred onto an output magnetic tape file or files and the input and output data require more than 1000 characters for storage. Records up to 1000 characters in size can be handled during a FORMAT operation. Records up to 3000 characters in size can be handled by SUBSET. Because the space available in MADAM program storage is limited, the actions provided in PRINT and FORMAT for printing and for textual processing have been sacrificed in SUBSET in order to provide for larger output records.

The SUBSET operation is initiated by a declaration in the form:

SUBSET (output file designations) (input file designation--INPUT A)

Examples:

SUBSET FILE 1 TAPE 2 FROM FILE 2 TAPE 3

SUBSET 1,2 FROM 2,3

SUBSET 1,2 and 1,4 FROM 2,3

The input file to the SUBSET operation may be on magnetic tape or in card form. The output from a SUBSET operation may be punched into cards rather than written onto tape. In the case of punched card output, only the last (or right-most) 80 characters of the object field as specified in the WRITE or WRITB action will be transferred to cards. The SUBSET operation requires only a single input file. The data from the file designated as input A are read via a READA action into internal region INA commencing at position INA 0. During an ABSTRACT operation, the data from input B is placed in region INB in a similar manner. The action READB and the test "IF ENDB" relating to operation with a second input file, as in the ABSTRACT operation below, are not valid for the SUBSET operation. The region INB may be used during SUBSET, however, for working storage.

4. The ABSTRACT operation is used when two input files are to be matched and, or selectively processed such that records from either or a combination of both files go to make up the output file or files. This is usually accomplished by one input file serving as the index or selector of data from the other input file. As in SUBSET, textual processing and printing have been traded for data capacity of 3000 characters.

The ABSTRACT operation is initiated by a declaration in the form:

ABSTRACT (output file designations) (input file designation--
INPUT A) (index or selector file designation--INPUT B)

Examples:

ABSTRACT FILE 1 TAPE 4 FROM FILE 3 TAPE 3 BY FILE 1 TAPE 2

ABSTRACT 1,4,3,3,1,2

ABSTRACT 1,4 FROM 1,3 by CARDS

ABSTRACT 1,2 AND 1,4 FROM 1,3 BY CARDS

The names "input file" for input A and "index or selector file" for input B are arbitrary. During an ABSTRACT operation the data from either input A or input B may be employed as selector criteria, and furthermore data from either A or B may be selected for output.

The "input file" or input A is read via a READA action into region INA. End-of-file for input A is tested with an "IF ENDA" test. The "index or selector file" or input B is read via a READB action into region INB. End-of-file for input B is tested with an "IF ENDB" test. Either of the two input files may be in card form. The output may be written onto punched cards as in SUBSET.

SUBSET and ABSTRACT Actions

Actions that may be taken during a SUBSET or ABSTRACT operation are, for the most part, the same as those for PRINT and FORMAT. Only differences and exceptions are listed below.

The following actions in PRINT and FORMAT are excluded from use in SUBSET and ABSTRACT.

PRINT

REPORT

SKIP

SCAN

RESTORE

It can be seen that actions relating to direct printing and to textual processing are not provided. Data regions relating to these actions are also eliminated. The MADAM terms COLUMN, TERM, MORE, PAGE, LINE, WORD, SUFFIX, PREFIX, PRINT, REPORT, SKIP, RESTORE, and SCAN are all inoperative during a SUBSET or ABSTRACT operation.

The following regions may be referred to in the SUBSET and ABSTRACT operations.

<u>REGION</u>	<u>SIZE</u>	<u>CONTENT</u>
INA	1000 char.	Contains an input File A record as a result of a READA action. May be used for other storage as well.
INB	1000 char.	Contains an input File B record as a result of a READB action. May be used for other storage.
OUT	1000 char.	May be used to prepare output data records.
TEMP	100 char.	May be used for temporary storage. Contains index values for the PRIME operation.
RESULT	10 char.	Accumulates arithmetic results.
REMAINDER	10 char.	Contains remainder following a divide action (Operator /).

The four operations above have many functions in common, and therefore, utilize common language terms to effect control. For these operations, MADAM terms can be separated into five classes:

- (1) Action terms;
- (2) Condition terms;
- (3) Data terms;
- (4) Language structure terms;
- (5) Special terms.

Action Terms

An action term must have an object in order to complete its meaning. Objects of action terms are normally data field descriptions which are composed of data terms. For example, the action statement "SET IN 0, 25 AT OUT 20" has the action term SET, and the data terms IN, 0, 25, OUT and 20.

Data terms are normally grouped together to provide a complete description of a data field. In the example above, "IN 0, 25" identifies the data field in region IN starting at relative position 0 in that region and comprising 25 characters. "OUT 20" identifies relative position 20 in region OUT as the starting location of the destination field.

Actions that imply movement or transfer of data require both an object and a destination. ADJUST, ALIGN, and SET are transfer actions and require an object field description and destination description. The terms +, -, *, /, =, and COMPUTE specify arithmetic actions and require only data field descriptions as objects.

SCAN and WRITE or WRITB also require only a field description as their object. READ, READA, READB, PRINT, SKIP, REPORT, and RESTORE provide user control of data input and output operations where applicable. No object data field is required, although most of these actions are subject to finer definition by means of a numeric parameter. "SKIP 2," for example, causes two lines to be left blank on the printed output page.

The system terms ABSTRACT, BLOCK, COMBINE, COPY, FORMAT, LOAD, POSITION, PRINT, SORT, and SUBSET are in reality action terms requiring data file descriptions as objects.

Language Structure Terms

The action DO takes a label as its object. Labels are language structure terms that act to mark a particular place in a MADAM specification. The DO action is like a branch or jump instruction in a general purpose computer. As a result of the DO action, the listed sequence of MADAM tests and actions is terminated and taken up at the particular place in the specifications marked by the object label.

Labels are also used to mark the termination of a string of conditional actions. For this purpose a label has the same meaning as the word "otherwise" in normal discourse. Examples of the use of labels are given in section F.

Condition Terms

The term IF is a condition term. IF introduces a condition statement or test. Condition statements are followed immediately by actions to be performed if the condition is true. If the tested condition is false, the system continues operation at the next IF statement in sequence or the next label, as indicated above. Conditional expressions take the form: IF (object field) relationship (referent field).

Condition statements in this form may be strung together to construct complex expressions of multiple conditionality through the use of the terms AND and OR to replace the term IF in all but the first statement of the string.

Conditions connected by AND must each be satisfied for the related action to take place. Conditions connected by OR may alternatively be satisfied for the related action to be performed. For example, the statement "IF A and B or C and D" will be true when:

1. Both A and B are true regardless of C and D.
2. Both C and D are true regardless of A and B.

The statement is false for all other conditions.

The statement: "IF A or B and C or D" will be true when:

1. A is true regardless of B, C, and D.
2. D is true regardless of A, B, and C.
3. Both B and C are true regardless of A and D.

The statement is false for all other conditions.

Relationship is specified by use of the terms:

EQ for equal

NQ for not equal

LS for less than

GR for greater than

GQ for not less than (greater than or equal)

LQ for not greater than (less than or equal)

In subsequent discussions the field description preceding the relationship term is referred to as the "object field." The field description following the relationship term is called the "referent field." Examples of the use of condition statement are given in section F.

Special Terms

The terms ANY, MORE, NEXT, NUMERIC, WORD, END, ENDA, ENDB, SUFFIX, and PREFIX require special description.

SCAN, WORD, SUFFIX, and PREFIX are applicable to the PRINT and FORMAT operations only and provide user control of textual data handling. The term SCAN and its object describe the textual data field. The user may choose up to five alphanumeric characters that may be declared as word separators. The beginning and ending positions occupied by the textual field determine that field as if it were bounded by word separators. A word is a string of alphanumeric characters, preceded and followed by at least one word separator. A word must not exceed 132 characters in length. Words are scanned in sequence, commencing with the left-most or lowest numbered position of the textual field.

The term WORD refers to the word last scanned. The terms NEXT WORD refer to the word about to be scanned and cause the word to be operated upon.

The term PREFIX refers to all preceding characters in the text field up to but not including the current word.

The term SUFFIX refers to all following characters in the text field including the separator character that terminated the current word.

The term MORE may be used only as the object of a test. The condition tested by "IF MORE" is true when there are words remaining in the textual field that have not been scanned. Textual field processing will be described in more detail in subsequent paragraphs.

The term END when used in PRINT or FORMAT as the object of a test refers to the condition end-of-file for the input data. For example, "IF END STOP ELSE. DO A" expresses a test for end-of-input-file wherein if the condition is true, return is made to MADAM system control. If the condition is false, operation is continued at label A. Note that the term ELSE is followed by a period and is therefore a MADAM label. It is required to mark the termination of the string of actions to be taken if the tested condition is true.

The terms ENDA and ENDB have similar meaning for SUBSET and ABSTRACT operations and refer respectively to end-of-file for the input data File A and end-of-file for the index or selector data File B.

The term NUMERIC is used as a condition and specifies a test that is applied to each character of the object field. For example, "IF IN 0, EQ NUMERIC" would be true if every character in relative positions 0 thru 4 in region IN was a number. Note that NUMERIC may only be used in the referent field position; thus the statement "IF NUMERIC EQ IN 0." is illegal.

The term ANY may precede any object field specification that is a part of a condition statement. For instance, "IF ANY IN 0,5 EQ (2)" is a legal form. "IF (2) EQ ANY IN 0,5" while logically correct is not properly translated by the current MADAM System. The presence of the term ANY has two effects on the test that will be performed.

- (1) The object field will be considered as a configuration of characters rather than as a value. Thus, characters that are not correctly differentiated in value comparisons, such as zero and blank, will be tested.
- (2) The referent field as a group of characters of given size will be compared with successive groups of equal size in the object field.

In the example, IF ANY IN 0,5 EQ (2), if the number 2 occupies any position in the object field IN 0,5 the condition will be true. The test "IF ANY (02468) EQ IN 0,1" will be true only if IN 0,1 is an even number.

Conditional tests modified by the term ANY, where object and referent field are of equal size, are executed by the MADAM System more rapidly than normal value comparison tests.

The term NEXT may be used in a data field description to replace the relative position term. For example, "SET IN 0,5 at OUT 0 SET IN NEXT, 5 at OUT NEXT" specifies transfer of two adjacent fields from region IN to adjacent positions in region OUT. Use of the term NEXT in the above context is valid for input and output regions only. Furthermore, the term NEXT refers to different data fields when used in connection with different specific input and output regions. The destination of a given NEXT transfer is relative to the character position in the specific destination region which has been established by previous transfer operations. The following illustrates this complex notion:

SET IN 50,5 at OUT 25

SET IN NEXT,5 at OUT NEXT

Each NEXT in this example is independently defined and represents totally different data strings. "SET IN 50,5 at OUT 25" defines "IN NEXT" as the object field IN 55. It defines OUT NEXT to be the destination location OUT 30. "SET IN NEXT,5 at OUT NEXT" actually transfers IN 55,5 to OUT 30. Now, since NEXT is dynamically redefined with each operation, IN NEXT has become IN 60, and OUT NEXT has become OUT 35.

The use of NEXT to declare string processes should be carefully proofed to avoid data region overflow. No system safeguards against such overflow are provided.

Data Terms

The form of a data term has been introduced previously in the example "SET IN 0,5 at OUT 0." A data field description is composed of three or fewer terms. The first of these is a region name. In the above example the term IN is a region name referring to the data input region. The second data term is a relative position in the named region occupied by the left-most character of the field. The above described field starts at relative position 0, the first position, in region IN. The third data term specifies the number of character positions comprising the field. In the example, the described field has five characters or numbers. The data field description OUT 0 does not have a third term as it conveys location information only. Destination descriptions do not require a third (size) term.

Often a user wishes to refer to a constant value in place of a data field. An alternative form is provided for such literal data entities. Any string of alphanumeric characters enclosed in parentheses is interpreted as a literal value. An example above, "IF ANY (02468) EQ IN 0,1" illustrates the use of literal (02468).

Some examples of data field specification are given below.

INA 10,5

The string of sequential characters in the region labelled INA, the input data region for ABSTRACT and SUBSET, beginning at relative position 10 in that region and consisting of five characters

INB 500,100

The string of sequential characters in the region labelled INB, the index or selector data region for ABSTRACT and SUBSET, beginning at relative position 500 in that region and consisting of 100 characters.

OUT NEXT 5

The string of five sequential characters immediately following the field previously specified in region OUT.

The first relative position in any region is 0. Fields may not exceed 132 characters in length.

Dynamic and Indirect Addressing

In more complex data manipulation exercises, a great saving in volume of specification can be effected by the generation of object data addresses dynamically and by iterative looping. In the MADAM System these functions are accommodated via the prime operation in conjunction with the operator DO.

You will recall that data fields are specified by two or three data terms. The first term is a region name. The second and third terms are numbers specifying the relative start location and size, respectively, of a data field. If either or both of the number terms of a data field description are followed immediately (with no intervening blank or comma) by a "prime" character (4-8 punch), the numbers are interpreted as relative addresses referring to 3 digit fields in region TEMP. The numeric data in region TEMP at the relative address specified is used as a relative start or size parameter. For example, if region TEMP contains the value "011006" in relative positions TEMP 0,6 the action "SET IN 0', 3' AT OUT 0'" would in effect "SET IN 11,6 AT OUT 11."* The values in region TEMP are numeric and fully manipulatable through the arithmetic operators of MADAM.

The prime operator is invalid for use with data regions other than IN, INA, INB, OUT, and TEMP. Any other usage will result in an error message and abort of translation.

Implicit Terms

For data field descriptions, when the region name is omitted, region IN for PRINT or FORMAT or region INA for ABSTRACT or SUBSET is assumed with the following exceptions.

- (1) For a destination description like OUT 0, region OUT is assumed.
- (2) For the action =, region OUT is assumed.

The PRINT operation assumes 60 lines per page if no REPORT action takes place.

If no test for input end-of-file is made in a specification, automatic termination will occur on the first input end-of-file. If, in an ABSTRACT specification, test is made for end-of-file input A and no corresponding test made for input B, automatic termination will occur on end-of-file input B. If the user provides for both end-of-file tests, he must then terminate his operation via a STOP action.

* Anyone familiar with index register usage in a general purpose computer will note that the effect of the PRIME operation is to provide up to 33 independent index registers for SUBSET and ABSTRACT and up to 66 such registers for PRINT and FORMAT.

During the FORMAT operation no carriage control actions will be taken unless a PRINT action has been used in the specification.

Special MADAM Terms

At the end of any set of operation specifications--after the END statement, for all but COPY and COMBINE--and before the operation declaration initiating the next set of specifications, the system will accept a control card with **** followed by a message to be logged on the printer. Since the usual purpose of such a message is to direct the computer operator to perform some support task, the system will halt. Operation will resume when the operator presses START. Between specifications as defined above, any text string on any number of cards may be excluded from consideration by the translation process by preceding and following the text string with a pair of sequential asterisks ** preceded and followed by at least one blank.

IV. USING THE MADAM SYSTEM

A. File Building

There are several MADAM operations applicable to file building. The LOAD operation is provided to optimize certain standard file building processes while minimizing user communication. Where the nature of the data or file structure is such that LOAD cannot be effectively employed, the operations FORMAT and/or SUBSET may be used. In FORMAT and SUBSET operations, the user must describe the file building process in detailed procedural form, expressing each conditional test and related transfer action necessary. He would choose between FORMAT and SUBSET on a basis of file structural complexity versus record size. The FORMAT operation, due to its greater number of possible conditional operations, can handle complex structures. The SUBSET operation makes available greater internal data storage space at the expense of some limitation in flexibility.

B. File Update

The process of file updating involves merging data from an update source into file records already in existence. Records in the existent file may be created, changed, or deleted. The ABSTRACT operation is provided for this purpose. Data from two input files can be combined in any logical or arithmetic way for inclusion in one or two output files.

C. File Organization

The SORT operation is provided to permit the records of a file to be placed in ascending collating sequence on selected character sets. Standard 1401 collating sequence is used. This sequence is illustrated in Appendix A of this volume.

A descending collating sequence can be generated by first sorting into ascending sequence, then generating sort keys in inverted order of numeric value, appending these generated keys to each record, and finally sorting once again by the generated key. The SUBSET operation can be used for sort key generation. Other sequences can be produced in like fashion.

D. Selecting a Subset from a File

The SUBSET operation provides for the selection of a subset file on a basis of logical or arithmetic criteria. Where the criteria for selection are related to properties represented in an input file different from the one being subset, the ABSTRACT operation can be employed. The second input file would contain a list of identities to be selected from the object file. In the latter case, both input files to the ABSTRACT operation should be ordered similarly.

E. Aggregation of Records Within a File

Records within a file may be grouped together into a single record or into sequential records of manageable size, either arbitrarily or on a logical basis by the BLOCK operation. Aggregation may be desirable in order to conserve magnetic tape storage. Each physical record on magnetic tape is followed by three-fourth inch of unused tape. Larger records minimize unused tape. Aggregation on a logical basis may be used as part of the process of file building. Records having the same identity are grouped together in a single block, thus larger and fewer records need be sorted in the organization process.

Aggregation of Files into a Single File

Related data from several different files may be needed to produce a report. The required data would be selected from the originating files by SUBSET. Each selected record would be given an identity key as a logical basis for association with appropriate records from other files. The resulting subset files would be combined into a single file through use of the COMBINE operation. The final file would be ordered on the identity key, thus grouping associated records together. Finally, the PRINT operation would prepare the printed report.

G. Report Generation

The PRINT operation is provided for the preparation and printing of reports on the direct printer. Where other peripheral devices are employed for report generation, the FORMAT operation can prepare formatted records for input to these devices via magnetic tape.

Table 1 below contrasts the relative capabilities of the procedural operations in the MADAM System.

PROCEDURAL OPERATIONS	PRINT	FORMAT	SUBSET	ABSTRACT
Number of input files handled	1	1	1	2
Number of output files handled	None	2	2	2
Available internal working storage	1132	1132	3000	3000
Number of conditional tests possible	50	50	30	30
Number of actions possible	100	100	50	50
Printed output possible	Yes	Yes	No	No
Textual input possible	Yes	Yes	No	No
Size of temporary storage (region T1FF)	200	200	100	100

TABLE 1

Examples

A set of MADAM specifications used to prepare a key-word-in-context listing of personal document files is presented below. Each MADAM statement is explained in detail.

The input data for the first operation were in IBM punched card form. Card columns 1 thru 25 contain topic tags, in this case parts of the document titles. Card columns 61 and 62 contained alphabetic codes representing document classification groups. The first MADAM operation, a FORMAT, selected the data fields from cards that were to receive further processing, eliminated extra blank characters that may occur between topic tags, and inserted a special non-blank character that printed as a blank between each tag.

FORMAT 1,2 FROM CARDS

The FORMAT operation was chosen so textual scanning could be done. The output file was File 1 on unit 2. The input data were on cards.

SET (5 blanks) AT TERM 0

All five of the alternate word separator characters were specified to contain blanks. That is, all alphanumeric characters excepting blanks may constitute a word.

1. READ SCAN 0,25
SET 60,2 AT OUT 5
SET () AT OUT 9

This string of actions is labelled "1" so that return can be made to it. The READ action acquires the first input card. The action SCAN 0,25 specified region IN 0 through IN 25 as a textual field. The region label is implicit. The action SM 60,2 at OUT 5, again with the implicit region label IN, transfers the two character classification code from relative position IN 60 (column 1 of the data card becomes relative position

0 in region IN) to OUT 5 and OUT 6. The action SET () at OUT 9 transfers a blank to position OUT 9. This action is used solely to initialize the OUT NEXT location at OUT 10.

2. SET NEXT WORD AT OUT NEXT
SET () NEXT

This action has a dual purpose. First the MADAM terms NEXT WORD cause the next word in the textual field to be located. Second, the SET action transfers the word to the next position in region OUT. The action SET () NEXT places the blank printing special non-blank character following the word. The character used was a 12/5/8 punch.

IF MORE DO 2

This test and DO action causes the complete string of topic tags to be found and transferred as above. When there are no more words (the tested condition is false) the next test or label is taken, in this case label 3.

3. WRITE OUT 0,80
SET IN 100,80 at OUT 0
DO 1

Eighty characters beginning with the character at position OUT 0 are written onto file 1 on unit 2. The SET IN 100, 80 AT OUT 0 takes advantage of the fact that region IN 100 and up is not used in this operation and therefore contains blank characters. This clears the OUT region to blanks. The DO 1 action causes return for a subsequent input card.

END

The END term signals to the MADAM translator that a complete set of operation parameters have been prepared. The translator passes control to the operating program.

The above specification, upon completion of operation logged "END OF FILE TAPE 0" on the line printer signifying that an end of file card (12/7/3 punch in column 1) had been read.

The following specified MADAM operation, again a FORMAT so that textual data fields could be handled, followed immediately after the previous operation with no operator intervention required. This second operation performed the actual key-word-in-context permuting of topic tags. For each topic tag, a record was written, formatted such that topic tags occupied the same relative position in each output record, and preceding and/or following words retained their proper position. To guarantee a proper sort order of topic tags (without following words effecting the order) each topic tag was duplicated as the trailing data field in the record.

FORMAT 1,3 FROM 1,2

File 1 unit 2, the output file from the previous process, became the input file to this. Output was written onto file 1 unit 3.

SET (4 blanks and a special character) AT TERM 0.

The character used was a 12/5/8 punch.

Four of the alternate word separators were set to blank and one was set to the special character placed by the previous process.

1. READ SCAN 10,36

The prior process placed the textual data at OUT 10. That location became IN 10 in this operation. Thirty-six characters were included in the textual field to permit this specification to be used as an initial process for specially formatted input cards having 36 character textual fields.

2. SET NEXT WORD AT OUT 25
SET SUFFIX NEXT
ADJUST PREFIX AT OUT 24

The action SET NEXT WORD AT OUT 25 acquired a word from IN 10 thru IN 45 and positioned it so that its leading character occupied OUT 25 and following. The action SET SUFFIX NEXT transferred all characters of the textual field that followed the particular word to adjacent positions so as to maintain trailing context. The action ADJUST PREFIX AT OUT 24, because of the non-blank special characters between words, transferred all of the textual field that preceded the particular word, positioning it so its last character,

In this case a blank-printing special character, adjoined the particular word on the left, maintaining preceding context.

SET 5,2 AT OUT 62
SET WORD AT OUT 64

The action SET 5,2 AT OUT 62 transferred the classification code to position OUT 62 and OUT 63. The next action set the topic tag at OUT 64 and following as a sort key.

WRITE OUT 0,80
SET IN 100,80 AT OUT 0

The resulting record as it was configured in region OUT was written onto file 1 unit 3 and the OUT region cleared to blanks as before.

IF MORE DO 2

An output record for each topic tag was assured. After the last topic tag in a textual field, label 3 was taken.

3. DO 1

The string of actions from label 1 thru label 3 was performed for each record from file 1 unit 2 until end of file on unit 2.

END

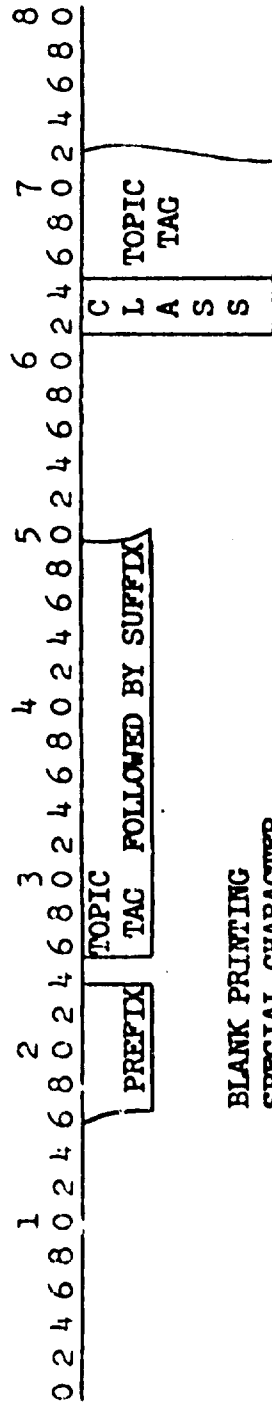
END signaled end of translation and start of operation as above.

No specified end of file test was required in either of the above operations. In both cases, the operation was terminated automatically by the system when end of file occurred on the input data file. In the last operation, "END OF FILE TAPE 2" was logged on the line printer.

File 1 on unit 3 now contained key-word-in-context data records but they were arranged in order of occurrence on topic tags. The next specification, again executed without intervention, caused rearrangement into alphabetic sequence of topic tags.

SORT 1,3 BY 64,10,80 PAUSE END The sort key placed at OUT by the previous specification was used for ordering. Ten characters were considered sufficient for proper order.

OUTPUT DATA FROM SECOND OPERATION
INPUT TO THE FIRST SORT AND HALF-LINE PROCESS



BLANK PRINTING
SPECIAL CHARACTER

At the completion of the first, or distribution pass of the SORT operation the message, "PASS 1 COMPLETE. REPLACE INPUT REEL IF DESIRED, CONTINUE" was logged on the direct printer and the system came to a programmed halt. The operator pressed the start button without saving the input data reel. File unit 3 was written over by the subsequent passes of SORT and ultimately became the alphabetically ordered data file.

At the completion of the SORT operation, "SORT COMPLETE, 009 PASSES" was logged and the following specifications were accepted without further intervention. This operation generated reordering keys to provide half-page formatting for the printed output of 56 lines per page. It assigned odd sequence numbers starting with "1" to subsequent records until the number "111" had been assigned at which time it commenced assigning even sequence numbers starting with "2." When the even number "112" had been assigned, a page number was incremented and the process repeated for all records in the file.

FORMAT 1,2 FROM 1,3

The previously ordered data file became input to this process. File 1 unit 2 was written over for output.

1. COMPUTE TEMP 0,3 + (1) =
TEMP 0,3

A page sequence number was established in TEMP 0. It was initialized at the value "1."

2. COMPUTE TEMP 3,3 + (1) =
TEMP 3,3

A record or half-line sequence number was established at TEMP 3 and initialized at "1."

READ

An input record was acquired in region IN commencing at IN 0. ,

SET TEMP 0,6 AT IN 64

The current value of page and half-line number was set into the input record at the position previously occupied by the topic tag sort key.

WRITE IN 0,70

Seventy characters were written onto file 1 unit 2 from region IN 0 thru IN 69.

COMPUTE TEMP 3,3 + (1) =
TEMP 3,3

The half-line sequence number was incremented by one. At this point, the number was even during the odd number assignment sequence and odd during the even sequence.

[illegible]

ХИАНН ИО САНУТИОА МВ.

**BLACK PRINTING
SPECIAL CHARACTER**

Pris: 1,-

IF TEMP 3,3 EQ (112)

SET (001) AT TEMP 3 DO 2

3. IF TEMP 3,3 EQ (113)

SET (000) AT TEMP 3 DO 1

Advantage was taken of this to establish tests for end of odd and even cycles. At the end of the odd cycle TEMP 3,3 contained "112" and reinitializing at "1" caused the sequence number "2" to be assigned first at label 2. At the end of the even cycle TEMP 3,3 contained "113" and reinitializing at "0" re-established odd number generation at step 1. Note that label 3 is required to mark the end of actions to be taken when the preceding test is true.

4. DO 2

Label 4 marks the end of actions to be taken when the test labelled 3 is true, label 4 constitutes, in effect, a statement "otherwise."

END

END marks the end of this specification.

"END OF FILE TAPE 3" was logged on the printer at the end of this operation. The resulting sequence-numbered file on unit 2 was now sorted into sequence number order.

SORT 1,2 BY 64,6 END specified this.

"SORT COMPLETE. 002 PASSES" was logged at the end of this sort. Only two passes were taken as the data file required only simple merging of odd and even records.

The next and last specification formatted the final printed output.

PRINT 1,2

The PRINT operation was chosen. The ordered file 1 on unit 2 contained the input data.

SET (02468) AT TEMP 10
READ

A character string representing all possible even digits was stored at TEMP 10 in preparation for odd-even testing. An initial record was read from file 1, unit 2 into region IN at IN 0 and following.

1. SET 25,3 AT TEMP 0

The first three characters of the first topic tag on each page were saved at TEMP 0 thru TEMP 2. These were used later at step 5 to provide alphabetic page content reference.

2. SET 0,64 AT 0

Both region IN and region OUT were implied. The action statement SET IN 0,64 AT OUT 0 would be equally appropriate. This action set the first half-line at position 0 in the print image.

READ IF END PRINT DO 5

The record containing the next half-line was read into region IN at IN 0. A test was made to see if end-of-file had been reached. If so, the print image containing the first half-line only was printed and operation taken up at step 5 where the page reference data line was prepared.

2A. SET 25,3 AT TEMP 3

The first three characters of each successive topic tag after the first of any page were saved at TEMP 3 thru TEMP 5. These provided a terminal alphabetic content reference for the page.

IF ANY TEMP 10,5 EQ IN 69,1

No label was required to terminate the previous action string since it was followed by a test. This test will be true if the single character at IN 69, the last digit of the half-line sequence number, is even. You will recall that a literal value (02468) was placed at TEMP 10 earlier.

SET 0,64 AT 67 READ

If this half-line had an even sequence number it was a second half-line and was placed in the print image at print position 67. Another record was read into region IN at IN 0.

IF END PRINT DO 5

Again a check was made for end-of-file. As before, if there were no more lines to be printed, operation was taken up at step 5. Observe that this test will be made whether or not the half-line was even numbered.

3. PRINT

The data in the print image was printed here when there had, as yet, been no end-of-file. The print image could have only a first half-line or it could have both half-lines.

IF LINE 0,3 EQ (57) DO 5

After the 56th line had been printed, the line counter, LINE 0,3 contained a value "57." If this page was complete, step 5, where a page reference line was prepared, was taken.

4. DO 2

If 56 lines had not yet been printed, return was made to step 2 where the current record in region IN was processed as a first half-line.

5. SKIP 2

Two lines were left blank to permit the page reference to stand out from the data lines.

SET (-) AT 64
SET PAGE 0,3 AT 65
SET (-) AT 68

The page number in the internal counter PAGE 0,3 was placed in the reference line image preceded and followed by a dash.

SET TEMP 0,3 AT 125
SET (-) AT 128
SET TEMP 3,3 AT 129

The previously stored page starting and ending alphabetic reference characters were placed in the reference line image separated by a dash.

PRINT RESTORE

The last or reference line for the page was printed and the printer carriage restored to print at the top of the next page.

IF END STOP

If step 5 had been taken by reason of end-of-file, this test caused termination of printing and return to MADAM System control.

6. DO 1

If there were records remaining to be printed, return was made to step 1 where page formatting began.

END

END terminated the translation of the PRINT specifications and initiated operation.

The set of MADAM specifications outlined above illustrate a use of the textual field processing operation as well as simple applications of arithmetic actions, the proper use of labels, elementary print actions, transfer actions, and DO actions.

V. MADAM SYSTEM COMPUTER OPERATIONS

The MADAM System Version IV is maintained in the form of a master magnetic tape. This MADAM MASTER tape contains a system generator program, as the first record, followed by the programs of the MADAM System as AUTOCODER output symbolic card records terminated by a tape mark (end-of-file). The MADAM MASTER tape is stored in a file-protected condition (the write ring is removed from the tape reel).

The MADAM Operating System tape is generated from the MADAM MASTER tape by the following actions:

- (1) Place the MADAM MASTER on unit 1 rewound to load-point.
- (2) Place a suitable blank or pool tape on unit 2 rewound to load-point.
- (3) Reset at the console, press LOAD FROM TAPE.

The MADAM Operating System will be generated on unit 2. Both reels will rewind at the end of the operation.

The MADAM Operating System tape operates from tape unit 1.

- (4) Dial unit 2 to unit 1.
- (5) Remove the MADAM MASTER and re-assign its tape unit.

From this point, MADAM Version IV System operations are similar to previous versions.

The MADAM MASTER tape described above may be copied onto another reel via the standard MADAM function COPY.

The system is called into operation initially by a LOAD FROM TAPE action at the computer console. Specification cards, and data cards if used, are loaded in the card reader. Appropriate data files on magnetic tape are loaded on the specific tape units corresponding to specification card instructions. The system's first action is to read a specification card. All subsequent system operations are directed by cards.


Operation specifications that are stacked in the card reader will be run continuously without interruption. When the card reader becomes empty, the system logs "CARD READER EMPTY" and pauses ready to read the next card. (Due to the nature of the IBM 1401, it is convenient to include two blank cards following the last specification card. These cards permit all control cards to be read before the card reader becomes inoperative due to an empty read hopper.)

To permit some operator control of diagnostic process and functional messages, sense switch options are provided as follows:

SENSE SWITCHMEANING

A	Always on to permit card reader empty to be sensed by the system.
B	On - Returns to try 10 more times to read a bad tape record after error halt. Off- Bypasses bad tape record after error halt.
C & D	Not assigned.
E	On - Printing of the first card of a specification is suppressed.
F	On - Printing of all but the first card of a specification is suppressed.
G	On - Core memory is dumped via the direct printer following the execution of each specification.

APPENDIX A

<u>PRINTS</u> <u>AS</u>	<u>DEFINED CHARACTER</u>	<u>CARD</u> <u>CODE</u>
(blank)	(blank)	
.	.	12-3-8
*) or		12-4-8
(blank)	(blank) or (12-5-8
(blank)	(blank) or <	12-6-8
** (blank)	≡ Group Mark	12-7-8
* + or	&	12
\$	\$	11-3-8
*	*	11-4-8
(blank)	(blank) or)	11-5-8
(blank)	(blank) or ;	11-6-8
(blank)	(blank) or ^	11-7-8
-	-	11
/	/	0-1
,	,	0-3-8
* (or	%	0-4-8
(blank)	(blank) or =	0-5-8
(blank)	(blank) or '	0-6-8
(blank)	(blank) or "	0-7-8
(blank)	¢	
* " or	#	3-8
* ' or	@	4-8
(blank)	(blank) or :	5-8
(blank)	(blank) or >	6-8
** (blank)	✓ Tape Mark	7-8
0 or +	? (Plus Zero)	12-0
A	A	12-1

(Continued)

2 December 1965

64

TM-2198/002/00

<u>PRINTS</u> <u>AS</u>	<u>DEFINED CHARACTER</u>	<u>CARD</u> <u>CODE</u>
B	B	12-2
C	C	12-3
D	D	12-4
E	E	12-5
F	F	12-6
G	G	12-7
H	H	12-8
I	I	12-9
-	! (Minus Zero)	11-0
J	J	11-1
K	K	11-2
L	L	11-3
M	M	11-4
N	N	11-5
O	O	11-6
P	P	11-7
Q	Q	11-8
R	R	11-9
** ±	± Record Mark	0-2-8
S	S	0-2
T	T	0-3
U	U	0-4
V	V	0-5
W	W	0-6
X	X	0-7
Y	Y	0-8
Z	Z	0-9
0	0	0
1	1	1

(Continued)

2 December 1965

65
(Last Page)

TM-2198/002/00

2

2

2

3

3

3

4

4

4

5

5

5

6

6

6

7

7

7

8

8

8

9

9

9

*Alternate character on some print chains.

**Characters have special significance for MADAM operation.